



Dataset of APT Persistence Techniques on Windows Platforms Mapped to the MITRE ATT&CK Framework

1st Khaled Rahal 
Cyber Defence Lab
Royal Military Academy
Brussels, Belgium

2nd Arbia Riahi 
Cyber Defence Lab
Royal Military Academy
Brussels, Belgium

3rd Thibault Debatty
Cyber Defence Lab
Royal Military Academy
Brussels, Belgium
t.debatty@cylab.be

Abstract—Securing against intrusions is a crucial aspect of cybersecurity defense. As systems become more complex and new technologies are introduced, new threats are constantly emerging, putting all aspects of information systems at risk. To detect these threats and put in place effective response strategies, security professionals must test their solutions on real-world data. This underscores the importance of datasets to provide a simulation of attack scenarios. Advanced Persistent Threats (APT) carry out multi-stage attacks on an organization’s network, often spanning extended periods. Existing datasets on APT mainly focus on the different kill chain stages such as initial intrusion, privilege escalation, lateral movements, and command and control. However, the persistence phase, which is crucial for the sustainability of attacks, is often neglected [10]. In this work, we propose a dataset specifically dedicated to the persistence techniques employed by the threat actor targeting the Windows platform. Our work offers a detailed analysis of persistence mechanisms, relying on realistic virtualized environments and attack simulation tools, based on MITRE ATT&CK TTP (Tactics, Techniques, and Procedures) used by known APT groups. Publicly available at [26], the datasets include detailed instructions for access and use, ensuring reproducibility and usability for researchers and cybersecurity practitioners.

Index Terms—Persistence Techniques, Advanced Persistent Threat, Mitre Att&ck, Datasets

I. INTRODUCTION

As systems become increasingly complex and new technologies are introduced, the defense against intrusions becomes a fundamental pillar of cybersecurity. Then, the threat landscape is continually evolving, and new risks are emerging to compromise every aspect of organization’s information infrastructure, leading to significant financial losses, from \$9.22 trillion to 2024 and \$13.82 trillion by 2028 [1]. APT represent a type of sophisticated cyberattacks orchestrated by attacker groups often operating under state sponsorship, but could also be financed by a company for competitiveness reasons. Their main goals include espionage, financial gain, intelligently property theft, sabotage, etc. Consequently, APT pose significant challenges to organizations by employing persistence techniques to maintain access to compromised systems over long periods. In particular, these techniques are essential to APT strategies, enabling attackers to survive system reboots, updates, and even some defensive

interventions. Their complexity and variety, especially in the Windows environment, make detecting and mitigating them a significant challenge for security professionals. While the MITRE [3] framework provides a comprehensive taxonomy of persistence techniques and sub-techniques, There is a significant gap in publicly available datasets that specifically focus on how APT manifest in real-world Windows systems. [10]. Addressing this gap, our work aims to analyze APT persistence techniques to create a new dedicated dataset. The main contribution of this paper is fourfold:

- Creation of a comprehensive dataset centered around the persistence techniques used by APT.
- Detailed Analysis of Persistence techniques employed by APT, providing a deeper understanding of how these threats sustain their foothold in target environments.
- Utilization of multiple attack tools (atomic red team, metasploit, caldera, etc.) to simulate a variety of persistence techniques in controlled environments, to ensure a broad coverage of potential APT tactics; and enhance the realism and applicability of the dataset.
- YAML-based configuration containing system versions, software details, and ground truth for persistence attacks to enhance the reproducibility of virtual machines.

Although the proposed dataset focuses primarily on the persistence techniques employed by APT groups, it also provides valuable insights for enhancing cybersecurity defenses and enables other researchers and practitioners to reproduce or extend our work for further studies or defensive measures. By analyzing the actions and tactics of these threat actors, security teams can use it to enhance their detection systems, improve threat intelligence, and refine incident response plans. As a result, our dataset serves as an essential tool for building more robust defenses against APT attacks.

This paper is organized as follows: Section II presents a literature review, summarizing existing research and datasets related to APT. Section III provides an in-depth discussion of Windows persistence techniques. Section IV details the dataset design process, outlines the installation procedure, and

discusses the tools used for setup, emulation of red team attacks, and data collection. Section V describes the simulation and validation methods employed to ensure the dataset’s reliability and applicability. Finally, the paper concludes with a summary of findings and potential future directions for research.

II. LITERATURE REVIEW

A. APT and Persistence Techniques

1) *APT*: A sophisticated, prolonged cyberattack in which an attacker quietly infiltrates a network, maintaining a hidden presence over an extended period to steal sensitive information. APTs are meticulously orchestrated, targeting specific organizations, bypassing security defenses. [2]. APT often exploit living-off-the-land binaries (LOLBins) that are pre-installed on Windows systems. These binaries are trusted by the system and are usually whitelisted by security solutions, making them ideal for bypassing detection mechanisms [4]. Persistence techniques are integral to APT strategies, enabling attackers to maintain long-term access to systems even after reboots or security interventions.

2) *Persistence Techniques*: According to MITRE [3], persistence techniques consist in access gained by adversaries to system across system restarts, credential changes, and other disruptions. Persistence techniques involve any actions, configurations, or access modifications that allow attackers to retain control over systems. These methods may include replacing or hijacking legitimate code or adding startup entries. Figure 1 illustrates the various stages of the cyber kill chain

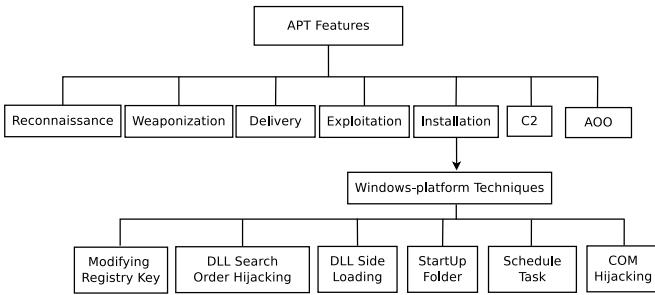


Fig. 1. APT feature taxonomy framework

[5] and highlights the related persistence techniques commonly used on Windows platforms, such as modifying registry keys, DLL hijacking, startup folder manipulation, scheduled tasks, and COM hijacking. The MITRE database has mapped these techniques comprehensively, as shown in figure 2, where key examples of persistence mechanisms employed by APT groups are presented.

Several groups, such as APT41, FIN13, APT29, APT28, and APT3, utilize multiple persistence techniques across different stages of an attack. This underscores the need for a dedicated dataset focused on persistence techniques. Understanding the variety of techniques employed is crucial for developing more effective detection and mitigation strategies.

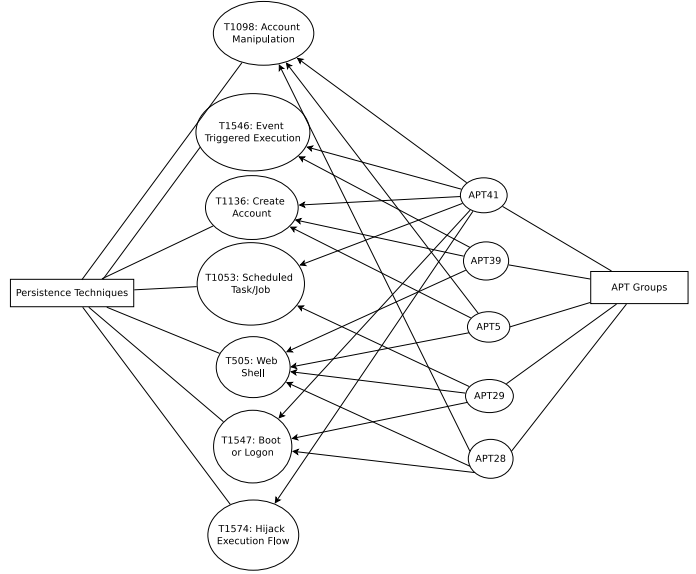


Fig. 2. Examples of persistence techniques used by APT groups

B. Related Works: Existing Datasets for APT

In this section, we explore existing datasets focused on APT, which are integral to advancing cybersecurity research and serve as the foundation for developing more effective detection systems. By comparing and analyzing the available datasets, we can identify gaps in APT research and demonstrate how our work contributes to addressing these critical voids. Several datasets have been developed to study the tactics, techniques, and procedures (TTPs) used by adversaries, capturing various stages of the kill chain, such as network infiltration, lateral movement, and data exfiltration. However, few resources provide detailed insights into the persistence techniques employed by APT groups. In this section, we review existing datasets related to APT behavior, emphasizing their strengths and limitations, particularly concerning persistence mechanisms.

The DAPT2020 dataset, introduced by Myneni et al. (2020) [7], covers four main stages of APT attacks: reconnaissance, foothold establishment, lateral movement, and data exfiltration. These stages involve a variety of attacks, such as network scans, brute force attempts, SQL injections, backdoors, denial of service (DoS), and privilege escalation. The dataset employs web shell technique for persistence and primarily focuses on network-based activities.

The SCVIC-APT-2021 dataset [9] models a multi-domain environment to simulate APT attacks. Covers several stages of APT attacks, including reconnaissance, initial compromise, lateral movement, pivoting, and data exfiltration. Unfortunately, it focuses on exploiting known CVEs, such as VSFTPD, and does not address any persistence mechanisms. Additionally, it emphasizes network-based activities.

Myneni et al. also developed a novel cybersecurity dataset called Unraveled [6], which provides a comprehensive view of APT attacks, addressing elements like attack planning,

deployment strategies, and attacker stealth. This dataset integrates APT attack information from reputable sources, such as MITRE's APT group database. For persistence, it is limited to the Hijack Execution Flow technique (path interception by modifying the environment variable) only.

The ATLASv2 dataset [11] captures realistic system logs by simulating both benign user behavior and attack scenarios. It includes attacks, such as exploiting known CVEs on victim machines, without utilizing persistence techniques [10].

Karim et al. [8] proposed a dataset that captures the Tactics, Techniques, and Procedures of APT attacks in Linux environments. Their dataset simulates various privilege escalation payloads for Linux, recent CVEs, and keylogger emulations. It also includes emulations of well-known APT groups such as APT41, APT28, APT29, and Turla, using persistence techniques like Account Manipulation, Create Account, and Web Shell. Although Karim et al. emulate APT groups in a Linux environment, their work lacks representation of Windows-based systems.

Although existing datasets are extensively focused on various attack techniques, they often overlook a crucial aspect-persistence mechanisms. Many attacks aim to maintain persistent access, existing datasets primarily focus on exploiting known vulnerabilities, without thoroughly addressing the persistence techniques commonly used in APT attacks. To the best of our knowledge, none of the previously presented dataset specifically focuses on persistence techniques.

Building on the insights gained from reviewing these datasets, we now focus on the specific persistence techniques employed on Windows, the most widely used operating system, accounting for approximately 73% of systems worldwide [25]. These techniques are essential for APT groups to maintain long-term control over compromised systems. Understanding them is crucial to enhance detection and defense strategies. In the following, we explore how these persistence techniques are utilized by APT groups in Windows environments, shedding light on their impact and effectiveness in real-world scenarios.

III. WINDOWS PERSISTENCE TECHNIQUES

A persistence technique allows an adversary to maintain access to a compromised machine by creating a backdoor, even after the system is rebooted. MITRE has documented 19 persistence techniques and 67 sub-techniques used by threat actor in Windows environments [3]. In this section, we focus on the most commonly used persistence (sub-)techniques [10]. Figure 3 illustrates the anatomy of persistence mechanism utilized in our proposed solution. We include a variety of techniques commonly exploited by attackers to maintain long-term access to compromised Windows environments. These techniques involve the creation and modification of Windows services, enabling the launch of malicious processes at system startup or specific intervals. We also investigate different DLL hijacking methods, such as DLL search order hijacking, side-

loading, which allow attackers to replace legitimate DLLs with malicious alternatives.

Furthermore, we integrate Windows Management Instrumentation (WMI) techniques that use Query Language to execute persistent scripts in response to specific system events. Scheduled tasks are another key component of our dataset, using tools like `schtasks.exe` and `powershell.exe` to create tasks that execute malicious programs at specified times or during system startup. Lastly, we incorporate changes to registry keys, in `HKEY_CURRENT_USER` and `HKEY_LOCAL_MACHINE`, to ensure that malicious applications are automatically launched during system boot or triggered by certain actions.

A. Registry Run keys /Startup Folder

Leveraging registry run keys or the startup folder are among the most methods used by malware and threat actors to achieve persistence. Figure 4 illustrates the behavior of Lokibot malware, which typically creates a payload file within hidden system folders, such as `AppData`.

In this instance, the payload is named `jkcgjj.vbs`. It is registered as a new value in the Run registry key, allowing it to execute automatically upon system startup. This ensures persistence and continued access to the compromised system.

B. Scheduled Task

The Windows Task Scheduler is a tool that enables users to automate tasks, executing them at specified times or in response to particular events. This includes launching programs, running scripts, and performing various actions without the need for human intervention. The `schtasks.exe` utility can be used from the command line or the Task Scheduler GUI with administrative privileges. APT29, use `schtasks` to create new tasks on remote hosts [18]. Additionally, they may modify existing legitimate tasks to execute their malicious tools. In our dataset we use `schtasks.exe` and `GhostTask` [19].

C. DLL Side-Loading

The Dynamic Link Library (DLL) is Microsoft's version of a shared library system. These libraries store code and data that multiple applications can access concurrently. The MuddyWater group uses DLL side-loading to maintain persistence on compromised networks, tricking legitimate programs into executing malicious payloads. Below, is an example of the attack chain involving DLL side-loading by MuddyWater [20]:

- The legitimate `GoogleUpdate.exe` loads the genuine `goopdate86.dll` module into memory.
- `goopdate86.dll` uses the DLL side-loading technique to load the malicious `goopdate.dll` into memory.
- The rogue `goopdate.dll` triggers `rundll32.exe` with the `DllRegisterServer` argument.
- `goopdate.dll` loads `goopdate.dat`, an obfuscated PowerShell script, into memory.
- The PowerShell script establishes communication with the Command and Control (C2) server.

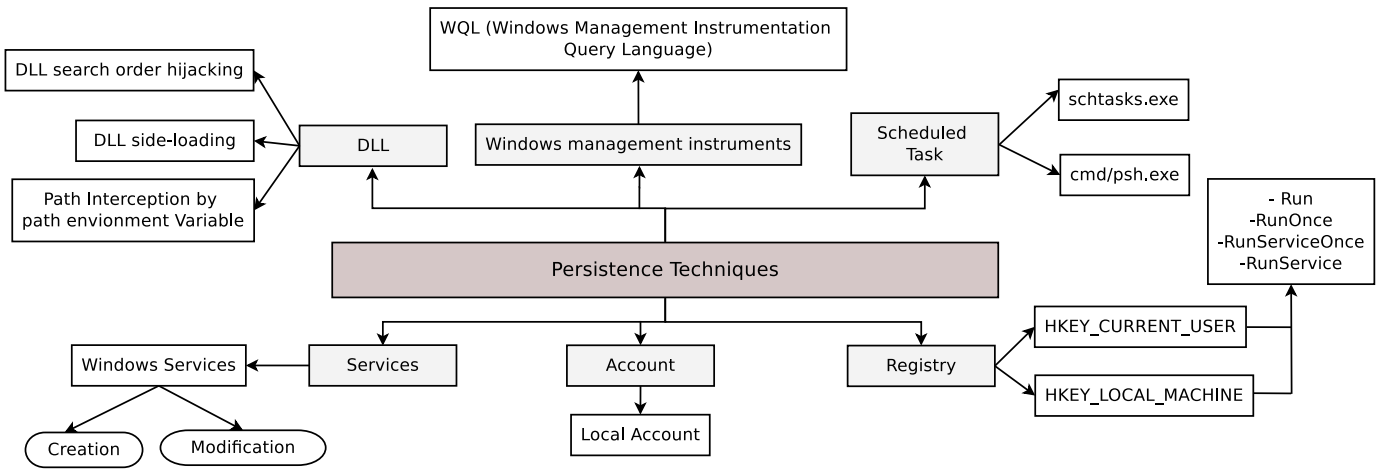


Fig. 3. Anatomy of persistence mechanism used in our dataset

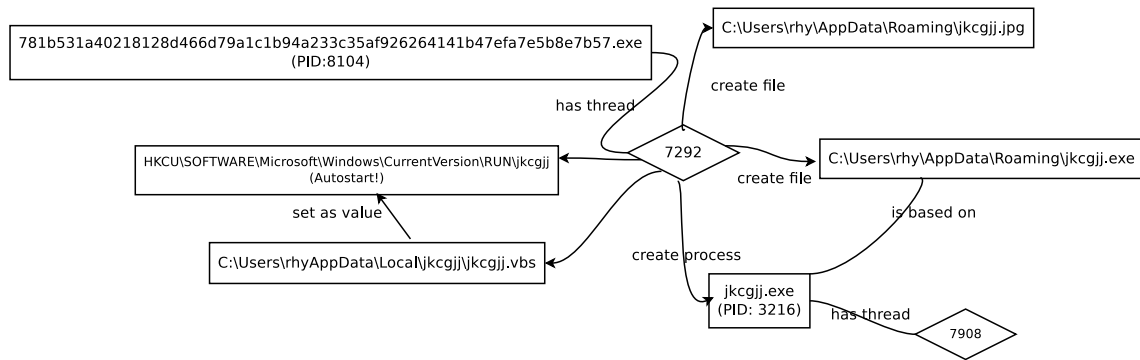


Fig. 4. LokiBot persists via Run registry [17]

D. Windows Service

Attacker create or modify Windows services to execute malicious payloads persistently. These services, which run in the background to perform essential system functions, are automatically started when Windows boots, allowing attackers to maintain their foothold on the system with each reboot. APT19 [22] utilizes this technique to maintain persistence within a compromised system.

E. Domain Accounts

An attacker creates a domain account to ensure long-term access to compromised systems. Managed by Active Directory Domain Services, this configuration sets access controls and permissions, allowing attackers to exploit shared resources and maintain persistence.

F. Windows Management Instrumentation (WMI) Event Subscription

WMI event subscription is a Windows feature that enables applications and scripts to receive notifications about system events. This management framework offers a standardized approach for administrators to query, monitor, and manage system resources efficiently. APT29 [24] has used WMI event subscriptions for persistence.

G. DLL Search Order Hijacking

Windows systems follow a specific sequence when searching for DLLs to load into programs. By exploiting this process, attackers can establish persistence, ensuring that their malicious code is loaded instead of legitimate DLLs. APT41 [23] use these techniques to execute malware by using benign and malicious code-signed Windows binaries.

IV. DATASET DESIGN

The dataset is designed to represent techniques employed by adversaries to maintain access to compromised systems across different stages of an attack. It encompasses a variety of persistence mechanisms observed in real-world scenarios.

A. System Setup

To build the dataset, we configured around 120 Windows virtual machines, including 85% benign systems and 15% infected ones. This setup mimics real-world scenarios and ensures comprehensive data collection, each virtual machines configured with specific user profile to simulate diverse environments. Each profile was customized with relevant software and configurations to accurately reflect the typical use cases of each role. Additionally, we connected all the virtual machines to Wazuh, an open-source platform for threat

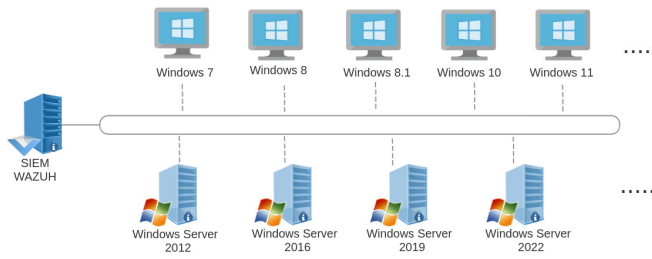


Fig. 5. Design architecture

detection and security monitoring according to predefined security rules [12]. Figure 5 illustrates the different Windows systems used in the creation of the dataset. Our focus is on persistence techniques at the endpoint level, rather than on network-related aspects.

1) *User Profile Simulation and Software Automation for Realistic Windows Environments*: To mimic real-world environments while creating the Windows machines, we designed distinct profiles for each system, representing different types of users including:

- Users,
- IT Administrators,
- Database Administrators,
- Security Specialist,
- DevOps Engineer,
- Developers,
- Financial Analyst,
- Designers,
- Video Editor,
- Researchers.

This diversity of profiles reflects the variety of users in real-world scenarios. For each profile, we installed unique software tailored to the specific needs of that user group. To automate the software installation process, we utilize Chocolateys [13], a powerful package manager for Windows, designed to simplify software installation and management, and ensure consistent and efficient deployment across all machines. Before installing the software, we first deployed Sysmon [16] on each machine, as it plays a crucial role in logging detailed system activity, such as process creation, network connections, file modifications, and registry changes. This level of monitoring is essential for detecting suspicious behavior and identifying persistence techniques used by adversaries. We also use mapped Sysmon events to the MITRE with a tailored schema. This configuration enables correlating specific Sysmon logs with MITRE techniques and activates various audit logs to ensure comprehensive data collection, including:

- Logon events,
- Object access,
- System events,
- Account logon,
- Process tracking,
- Policy change,

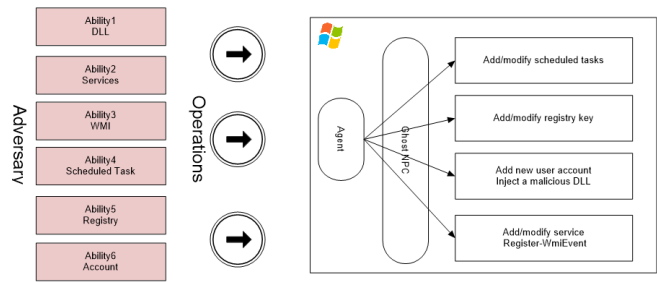


Fig. 6. Adversary operations and persistence techniques overview

- Privilege use,
- Directory service access,
- Account management.

The combination of Sysmon and audit logs enables capturing a wide range of system activities to provide a rich source of data for analyzing persistence techniques and security events. Additionally, we installed the Wazuh agent on each machine for monitoring purpose, as outlined in figure 5. It provides enhanced visibility and additional data [8], complementing the logs generated by Sysmon and the activated audit. This integration allowed us to capture a more comprehensive dataset, enriching the analysis of persistence techniques with Sysmon.

2) *Simulating APT Persistence Techniques Using Emulation Tools*: As illustrated in figure 6, we employed the Caldera platform [21], an open-source adversary emulation tool developed by MITRE, to simulate real-world attack scenarios. It allows automated execution of adversary behaviors based on the MITRE ATT&CK framework, making it ideal for testing and analyzing persistence techniques. By using Caldera, we were able to execute controlled attacks across the Windows machines, replicating APT tactics in a consistent and repeatable manner. This helped us to validate the effectiveness of the persistence techniques used and to ensure the dataset reflects authentic attack patterns. Besides, we integrated Metasploit with Caldera to add complexity to the attacks and create more realistic chaos.

Besides, we utilized Atomic Red Team [14], an open-source testing framework that provides a set of small, easily executable tests mapped to the MITRE ATT&CK framework. Atomic Red Team allowed us to simulate a wide range of persistence techniques across different versions of Windows. By executing atomic tests, we were able to verify the presence of various attack techniques, capturing detailed logs for analysis. This tool was useful to ensure that our dataset includes a broad spectrum of APT-like behaviors, making it more comprehensive and reflective of real-world attack patterns.

To generate realistic background noise, we integrated the GHOSTS [15] non-player character (NPC) Framework into our environment. It is an open-source framework designed to replicate user interactions with systems in a realistic and automated manner. This allowed us to test persistence

techniques in environments that closely resemble real-world usage patterns, and to validate the relevance and applicability of the dataset.

We manually collected logs from key sources, including Sysmon, Application, System, Security, and Task Scheduler files. This manual collection process closely mirrors real-world scenarios where centralized log collection systems may not always be available, offering a more realistic representation of how logs are gathered in different organizational settings.

B. Data Collection and Organization

We collected logs in EVTX format from Windows machines, capturing events from various system components. For each machine monitored by wazuh, we also gathered logs in JSON format generated by the wazuh agent. Additionally, we created a YAML file for each machine, containing a detailed description of the operating system and the installed software versions.

To facilitate reproducibility, we included a PowerShell script for automating the installation of the necessary software and configurations. The folder structure was organized systematically, as shown in Figure 7. For the malicious machines, we added a separate YAML file containing ground truth data, which will be used to detect malicious activities.

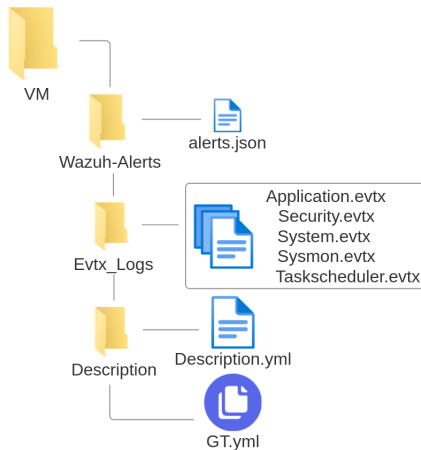


Fig. 7. Dataset folder Structure for logs, ground truth, and system components

V. SIMULATION AND VALIDATION

For our analysis, we selected two scenarios, each representing commonly observed APT behaviors. The techniques used in these scenarios, particularly persistence mechanisms, have been widely documented [3] as part of the tactics employed by several APT groups.

A. Scenario 1

As shown in Figure 8, the simulation included several techniques. It began with remote system discovery, enabling the identification of systems within the network. To establish persistence, techniques such as scheduled tasks and boot or logon autostart execution were employed, ensuring malicious

code execution during startup or user logon. Additional methods included creating Windows services and exploiting office application startup settings to execute malicious code upon application launch.

B. Scenario 2

We simulated APT29 attack using various techniques:

- 1) Execution and Discovery (Scripting): create a PowerShell script in the user's directory for execution.
- 2) Privilege Escalation (Sticky Keys Attack): modify system files like `osk.exe`, `sethc.exe`, and `displayswitch.exe` to gain privileged access using the Sticky Keys exploit.
- 3) Persistence:
 - Run Keys/Startup Folder: add an executable to the registry run keys, ensuring the malware runs automatically upon system startup.
 - Scheduled Task: create scheduled task to run a malicious update program at a specific time.
- 4) Credential Access (Keylogging): use powershell script to log keystrokes.
- 5) Additional Persistence:
 - WMI: register event to trigger the execution of malware.
 - Shortcut Modification: modify shortcuts to maintain persistence by adding them to the startup folder.
- 6) Defense Evasion:
 - File Deletion: Sdelete to clear temporary folders, erasing traces of malicious activity.
 - Use Rundll32: Execute malicious code via a DLL.

C. Validation

To validate the effectiveness of our dataset, we employed two attack simulation tools: Caldera and Atomic Red Team. Both tools are directly mapped to the MITRE framework, ensuring that the techniques used during validation align with the TTP employed by APT. We conducted simulation examples that demonstrate how the dataset successfully captures activities mapped to the MITRE framework.

1) *Registry Run Keys / Start Folder*: To validate the persistence technique T1547.001, we executed the command described in line 4 Table I to add a registry value that would trigger the execution of the specified DLL `updateall.dll` upon system startup.

The following event logs were captured, confirming the addition of the registry entry:

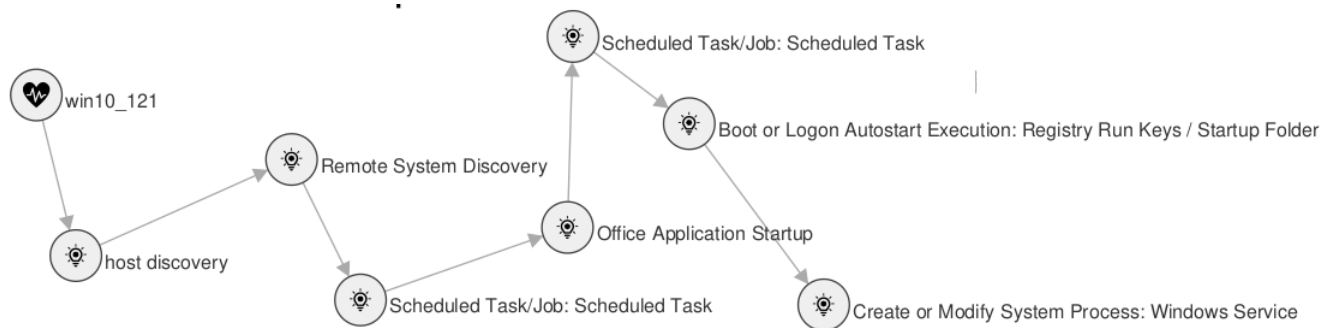


Fig. 8. Technique used in scenario 1 generated by caldera

N°	Command	Description
1	reg add "HKLM\Software\Microsoft\Windows NT\CurrentVersion\Image File Execution Options\osk.exe"	Bypass authentication with sticky keys exploit
2	reg add "HKCU\Software\Microsoft\Windows\CurrentVersion\Run" /v "Run key persistence" /t REG_SZ /d "C:\tmp\ghost.exe" /f	Add persistence through registry run keys
3	SCHTASKS /create /SC ONCE /TN "spawn" /TR "gosupdate.exe" /ST 18:00	Schedule a task for malicious updates
4	REG ADD HKLM\SOFTWARE\Microsoft\CurrentVersion\RunOnceEx\0001\Depend /v 1 /d "C:\tmp\updateall.dll"	Add registry value for DLL execution
5	sc.exe localhost create AtomicTestService_CMD binPath= "411da5.exe" start=auto type=Own && sc.exe localhost start AtomicTestService_CMD	Create and start a malicious service
6	schtasks /create /tn updatedaily /tr "C:\Windows\TEMP\stark.exe" /sc minute /mo 5	Schedule task to run malware repeatedly
7	GhostTask.exe localhost add updats "cmd.exe" "c notepad.exe" win10 weekly 11:11 monday	Create weekly task to launch Notepad.exe

TABLE I
EXAMPLES OF USED COMMANDS AND THEIR PURPOSES

Log Evidence - Registry Value Set

Registry value set:
RuleName:technique_id=T1547.001,
 technique_name=Registry Run Keys / Start Folder
EventType: SetValue
UtcTime: 2024-10-02 19:16:07.747
ProcessGuid:{30f6f5d4-9bf7-66fd-0505-000000000700}
ProcessId: 8400
Image: C:\Windows\system32\reg.exe
TargetObject: HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnceEx\0001\Depend\1
Details: C:\tmp\updateall.dll
User: DESKTOP-0B7RNLE\win10

Log Evidence

Registry value set:
RuleName: -
EventType: SetValue
UtcTime: 2024-10-02 19:18:59.794
ProcessGuid:{30f6f5d4-8a99-66fd-0b00-000000000700}
ProcessId: 628
Image: C:\Windows\system32\services.exe
TargetObject: HKLM\System\CurrentControlSet\Services\AtomicTestService_CMD\ImagePath
Details: 411da5.exe
User: NT AUTHORITY \SYSTEM

2) *Create or Modify System Process:* For technique T1543.003, we executed the command described in line 5 Table I to create a new service using 'sc.exe' called AtomicTestService_CMD and sets the executable path to 411da5.exe, which will be automatically started. The captured event confirms the alteration of The registry key associated with the newly created service. This log verifies that the service was created and successfully registered within the system:

3) *Scheduled Task/Job:* We ran the command outlined in line 6 of Table I to create a scheduled task (T1053.005) named updatedaily that runs every 5 minutes, executing a file stark.exe from the temporary directory. The task was successfully registered in the system, as seen from the file creation logs under C:\Windows\System32\Tasks\updatedaily, and the related executable stark.exe was launched as foreseen by the task scheduler.

Log Evidence - Task Creation

File created:

RuleName: -

UtcTime: 2024-09-22 18:52:08.376

ProcessGuid:{6a6c6f4d-6224-66f0-1b00-00000000d00}

ProcessId: 1316

Image: C:\Windows\system32\svchost.exe

TargetFilename: C:\Windows\System32\Tasks\updatedaily

CreationUtcTime: 2024-09-22 18:52:08.376

User: NT AUTHORITY \SYSTEM

Log Evidence - Task Execution

Task Scheduler launched action:

Command: "C:\Windows\TEMP\stark.exe"

TaskInstance:{6bd65884-7e1e-4c18-9639-9743b1f948d3}

Task Name: \updatedaily

CONCLUSION AND FUTURE WORK

Our work presents a novel approach for creating datasets that focus on persistence techniques to disrupt the prolonged activity of APT. By concentrating on these mechanisms, we aim to enhance the detection and mitigation of persistence strategies employed by APT, ultimately reducing their dwell time within compromised systems. This approach allows more targeted defense, improving the effectiveness of security solutions in real-world scenarios.

Our dataset contains a simulation of the most common persistence mechanisms mentioned by MITRE ATT&CK [3]. We evaluated these mechanisms across various Windows versions and configurations, providing the logs in their raw EVT-X format to enable greater flexibility for feature extraction.

In the future, we aim to preprocess the datasets by converting raw data into structured formats, performing feature extraction and engineering, and testing various AI models to measure the effectiveness of these datasets in detecting and classifying attack techniques. Additionally, we will study countermeasures used by APT, particularly focusing on how they may intentionally adjust their tactics to evade detection, with the goal of enhancing our detection capabilities. The datasets are publicly available at [26].

REFERENCES

- [1] Statista. "Cybercrime Expected To Skyrocket in Coming Years.", [Online]. Available: <https://www.statista.com/chart/28878/expected-cost-of-cybercrime-until-2027/>.
- [2] CrowdStrike. "What is an Advanced Persistent Threat?.", [Online]. Available: <https://www.crowdstrike.com/cybersecurity-101/advanced-persistent-threat-apt/>.
- [3] The MITRE Corporation. "MITRE ATT&CK.", [Online]. Available: <https://attack.mitre.org>.
- [4] LOLBAS Project. "Living Off The Land Binaries, Scripts and Libraries.", [Online]. Available: <https://lolbas-project.github.io/>.

- [5] Nikkiah Bahrami, Pooneh & Dehghantaha, Ali & Dargahi, Tooska & Parizi, Reza & Choo, Kim-Kwang Raymond & Haj Seyyed Javadi, Hamid. (2021). "Cyber Kill Chain-Based Taxonomy of Advanced Persistent Threat Actors: Analogy of Tactics, Techniques, and Procedures". *Journal of Information Processing Systems*. 15. 10.3745/JIPS.03.0126.
- [6] Sowmya Myneni, Kritshekhar Jha, Abdulhakim Sabur, Garima Agrawal, Yuli Deng, Ankur Chowdhary, Dijiang Huang. Unraveled — A semi-synthetic dataset for Advanced Persistent Threats. *Computer Networks*, Volume 227, 2023, 109688, ISSN 1389-1286.
- [7] Myneni, Sowmya, et al. "DAPT 2020-constructing a benchmark dataset for advanced persistent threats." *Deployable Machine Learning for Security Defense: First International Workshop, MLHat 2020, San Diego, CA, USA, August 24, 2020, Proceedings 1*. Springer International Publishing, 2020.
- [8] Karim, Syed Sohaib et al. (2024). "Advanced Persistent Threat (APT) and intrusion detection evaluation dataset for linux systems 2024." In: *Data in Brief*, p. 110290.
- [9] Jinxin Liu, Yu Shen, Murat Simsek, Burak Kantarci, Hussein T. Mouftah, Mehran Bagheri, Petar Djukic, June 20, 2022, "SCVIC-APT-2021", *IEEE Dataport*, doi: <https://dx.doi.org/10.21227/g2z5-ep97>.
- [10] Liu, Q., Shoaib, M., Rehman, M. U., Bao, K., Hagenmeyer, V., & Hassan, W. U. (2024). Accurate and Scalable Detection and Investigation of Cyber Persistence Threats. *arXiv preprint arXiv:2407.18832*.
- [11] A. Riddle, K. Westfall, and A. Bates. "ATLASv2.", [Online]. Available: <https://bitbucket.org/sts-lab/atlasv2/src/master/>.
- [12] Stanković, Stefan, Slavko Gajin, and Ranko Petrović. "A Review of Wazuh Tool Capabilities for Detecting Attacks Based on Log Analysis." *No Nama Agent Integrity File Added Delete Modified 1* (2022).
- [13] Chocolatey, [Online]. Available: <https://chocolatey.org/install>.
- [14] Okuma, Momoka, et al. "Automated Mapping Method for Sysmon Logs to ATT&CK Techniques by Leveraging Atomic Red Team." *2023 6th International Conference on Signal Processing and Information Security (ICSPIS)*. IEEE, 2023.
- [15] Georgi Nikolov. Simulate user activity with the GHOSTS framework: Client set-up and Timelines, [Online]. <https://cylab.be/blog/81/simulate-user-activity-with-the-ghosts-framework-client-set-up-and-timelines>.
- [16] Smiliotopoulos, Christos, Georgios Kambourakis, and Konstantia Barbatsalou. "On the detection of lateral movement through supervised machine learning and an open-source tool to create turnkey datasets from sysmon logs." *International Journal of Information Security* 22.6 (2023): 1893-1919.
- [17] Le, Tran Duc & Dinh, Duy & Nguyen T. H., Phuoc & Muthanna, Ammar & Abd El-Latif, Ahmed. (2023). Exploring Common Malware Persistence Techniques on Windows Operating Systems (OS) for Enhanced Cybersecurity Management. 10.1201/9781003369042-7.
- [18] Fortinet. "TeamCity Intrusion Saga: APT29 Suspected Among the Attackers Exploiting CVE-2023-42793", [Online] <https://www.fortinet.com/blog/threat-research/teamcity-intrusion-saga-apt29-suspected-exploiting-cve-2023-42793>
- [19] Ghost Scheduled Task, [Online] <https://github.com/netero1010/GhostTask>
- [20] TTPs and IOCs Used by MuddyWater APT Group in Latest Attack Campaign, [Online] <https://www.picussecurity.com/resource/blog/ttp-ioc-used-by-muddywater-apt-group-attacks>
- [21] Applebaum, Andy, et al. "Intelligent, automated red team emulation." *Proceedings of the 32nd annual conference on computer security applications*. 2016.
- [22] Analyzing APT19 malware using a step-by-step method, [Online] <https://cybergeeks.tech/analyzing-apt19-malware-using-a-step-by-step-method/>
- [23] APT41 Has Arisen From the DUST, [Online] <https://cloud.google.com/blog/topics/threat-intelligence/apt41-arisen-from-dust?hl=en>
- [24] Dissecting One of APT29's Fileless WMI and PowerShell Backdoors (POSHSPY), [Online] <https://cloud.google.com/blog/topics/threat-intelligence/dissecting-one-of-apt29?hl=en>.
- [25] Desktop Operating System Market Share Worldwide, [Online] <https://gs.statcounter.com/os-market-share/desktop/worldwide>
- [26] <https://gitlab.cylab.be/cylab/datasets/apt-persistence>